

(12) NACH DEM VERTRAG ÜBER DIE INTERNATIONALE ZUSAMMENARBEIT AUF DEM GEBIET DES  
PATENTWESENS (PCT) VERÖFFENTLICHTE INTERNATIONALE ANMELDUNG

(19) Weltorganisation für geistiges Eigentum  
Internationales Büro



(43) Internationales Veröffentlichungsdatum  
25. August 2005 (25.08.2005)

PCT

(10) Internationale Veröffentlichungsnummer  
**WO 2005/078599 A1**

(51) Internationale Patentklassifikation<sup>7</sup>: **G06F 15/80**,  
17/50

(21) Internationales Aktenzeichen: PCT/EP2005/050500

(22) Internationales Anmeldedatum:  
7. Februar 2005 (07.02.2005)

(25) Einreichungssprache: Deutsch

(26) Veröffentlichungssprache: Deutsch

(30) Angaben zur Priorität:  
10 2004 007 232.9  
13. Februar 2004 (13.02.2004) DE

(71) Anmelder (für alle Bestimmungsstaaten mit Ausnahme von  
US): **SIEMENS AKTIENGESELLSCHAFT** [DE/DE];  
Wittelsbacherplatz 2, 80333 München (DE).

(72) Erfinder; und

(75) Erfinder/Anmelder (nur für US): **WIEGAND, Christian** [DE/DE]; Bornhardtstr. 6, 38678 Clausthal (DE).  
**SIEMERS, Christian** [DE/DE]; Mary-Cassatt-Ring 38,  
38446 Wolfsburg (DE).

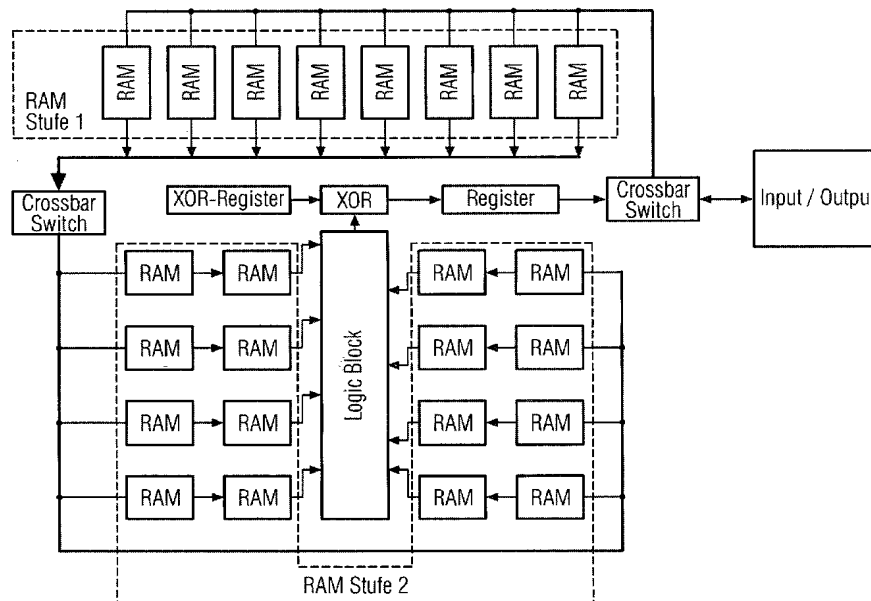
(74) Gemeinsamer Vertreter: **SIEMENS AKTIENGESELLSCHAFT**; Postfach 22 16 34, 80506 München (DE).

(81) Bestimmungsstaaten (soweit nicht anders angegeben, für  
jede verfügbare nationale Schutzrechtsart): AE, AG, AL,  
AM, AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH,  
CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES,  
FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE,  
KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD,  
MG, MK, MN, MW, MX, MZ, NA, NI, NO, NZ, OM, PG,  
PH, PL, PT, RO, RU, SC, SD, SE, SG, SK, SL, SY, TJ, TM,

[Fortsetzung auf der nächsten Seite]

(54) Title: RECONFIGURABLE SWITCHING DEVICE FOR PARALLEL CALCULATION OF ANY PARTICULAR ALGORITHMS

(54) Bezeichnung: REKONFIGURIERBARES SCHALTWERK ZUR PARALLELEN BERECHNUNG BELIEBIGER ALGORITHMEN



(57) Abstract: The invention relates to the configurable architecture of a calculation device comprising at least one individual configurable and/or reconfigurable switching device, whereby the output variables thereof form a time point  $t_{n-1}$  and the input variables of the switching device form a time point  $t_n$ . Means are provided in order to control the output variables in a clocked manner and to store the output variables of the switching device between the time points  $t_{n-1}$  and  $t_n$ .

[Fortsetzung auf der nächsten Seite]

WO 2005/078599 A1



TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, YU, ZA, ZM, ZW.

PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

**(84) Bestimmungsstaaten** (*soweit nicht anders angegeben, für jede verfügbare regionale Schutzrechtsart*): ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), eurasisches (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), europäisches (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IS, IT, LT, LU, MC, NL,

**Veröffentlicht:**

— mit internationalem Recherchenbericht

*Zur Erklärung der Zweibuchstaben-Codes und der anderen Abkürzungen wird auf die Erklärungen ("Guidance Notes on Codes and Abbreviations") am Anfang jeder regulären Ausgabe der PCT-Gazette verwiesen.*

---

**(57) Zusammenfassung:** Die konfigurierbare Architektur einer Rechneinrichtung weist wenigstens ein einzeln konfigurierbares und/oder rekonfigurierbares Schaltwerk auf, dessen Ausgangsvariablen zu einem Zeitpunkt  $t_{n-1}$  die Eingangsvariablen des Schaltwerks zu einem Zeitpunkt  $t_n$  bilden. Es sind Mittel zu einem taktge- steuerten Speichern der Ausgangsvariablen des Schaltwerks zwischen den Zeitpunkten  $t_{n-1}$  und  $t_n$  vorgesehen.

## Beschreibung

## REKONFIGURIERBARES SCHALTWERK ZUR PARALLELEN BERECHNUNG BELIEBIGER ALGORITHMEN

5

## 1 Einleitung

Die vorliegende Erfindung bezieht sich auf eine Architektur einer Rechneinrichtung zur parallelen Berechnung von Algorithmen mit wenigstens einem Schaltwerk.

## 2 Stand der Technik

## 2.1 Bekannte Rechnermodelle

15

Allgemein wird die sogenannte ‚Von-Neumann-Architektur‘, wie sie aus der prinzipiellen Darstellung der Figur 1 hervorgeht, als Universalrechneinrichtung angesehen. Darunter ist zu verstehen, dass mithilfe eines Rechners, auf dieser Architektur mit den Komponenten Central Processing Unit [CPU, darin enthalten Control Unit (CU) und Arithmetical-Logical Unit (ALU)], Memory (Speicher), Input/Output (Ein-/Ausgabe) und Bussystem basierend, alle algorithmierbaren Probleme prinzipiell berechnet werden können. Die Einstellung eines solchen Rechners auf das jeweilige Problem erfolgt durch ein Programm, d.h., eine textuelle Beschreibung des Algorithmus z.B. in den Programmiersprachen C, C++ oder Java. Dieses Programm wird durch einen Übersetzer (Compiler), selbst ein Programm darstellend, in ein Maschinen-lesbares Programm übersetzt.

30

Die Programmbearbeitung erfolgt in den Rechner, die auf der Von-Neumann-Architektur nach Figur 1 und ähnlichen Architekturen (z.B. einer Harvard- oder modifizierten Harvard-Architektur) basieren, prinzipiell auf sequenzielle Weise. Dies ist so zu verstehen, dass der Algorithmus, bestehend aus einer Menge von Maschinen-Instruktionen, dadurch bearbeitet wird, dass die erste Instruktion bekannt ist. Die einem Be-

fehl nachfolgende Instruktion ist entweder die im Speicher an der nächsten Adresse stehende (normaler Programmfluss), oder die letzte ausgeführte Instruktion war ein Sprungbefehl, der den aktuellen Programmstand an eine andere Stelle versetzt.

- 5 Das interne Register, das den aktuellen Programmstand speichert, wird mit ‚Program Counter‘ (PC) bezeichnet.

Dieses Prinzip der sequenziellen Bearbeitung bedeutet, dass sich zu einem Zeitpunkt genau eine Instruktion in der Bearbeitung befindet. Es wird als Von-Neumann-Prinzip bezeichnet. 10 Moderne Architekturen, die als RISC (Reduced Instruction-Set Computing), superskalar oder VLIW (Very Long Instruction Word) bezeichnet werden, führen zwar zu einem Zeitpunkt mehr als eine Instruktion aus; das Prinzip der Sequenzialität 15 bleibt jedoch erhalten. Insgesamt wird dieses Ausführungsprinzip als ‚zeit-sequenziell‘ (Computing in Time) bezeichnet, was andeutet, dass der Algorithmus Zeit benötigt.

Ein gänzlich anderes Prinzip der Programmbearbeitung ist in 20 programmierbaren Logikbausteinen [PLDs (Programmable Logic Devices), entsprechend Figur 2] bzw. deren bekanntester Implementierung, den FPGAs (Field-Programmable Gate Arrays), vorgesehen. Auch diese Architektur ist universell, d.h. für jedes algorithmierbare Problem einsetzbar. Die Programmierung 25 erfolgt hierbei so, dass elementare Recheneinheiten, meist auf Bit-Ebene definiert und daher als Logikelemente bezeichnet, in einem Netzwerk verschaltet werden. Diese Form der Programmierung wird meist als ‚Konfiguration‘ bezeichnet.

30 Die Programmbearbeitung in einem PLD erfolgt im Unterschied zum Von-Neumann-Prinzip mit maximaler Parallelität. Die gesamte Konfiguration eines Bausteins kann als eine einzige Instruktion – im Gegensatz zum Von-Neumann-Prinzip allerdings nicht fest definiert, sondern zusammengesetzt – aufgefasst 35 werden, die in einem Zyklus komplett bearbeitet wird. Die Zykluszeit, häufig mit einem externen Takt in Verbindung gebracht, ist dann von der Komplexität der Zusammensetzung ab-

hängig. Hierdurch kommt ein im Vergleich zu Prozessoren niedrigerer Takt zum Einsatz, der aber durch die Parallelität der Ausführung mehr als ausgeglichen wird. Dieses Prinzip der Ausführung wird als ‚Computing in Space‘ bezeichnet.

5

## 2.2 Deterministische endliche Automaten

Eines der wesentlichen Modelle zur Algorithmierung von Problemen sind deterministische endliche Automaten [DEAs, auch DFAs (deterministic finite automata)]. Diese werden in der Technik auch als ‚Finite State Machines‘ (FSMs gemäß der prinzipiellen Darstellung nach Figur 3) bezeichnet. Dieses Modell betrachtet das Problem als eine Folge von Zuständen mit definierten Übergangsfunktionen (Next State Decoder) zwischen diesen, abhängig von den Eingangswerten. Obwohl das Modell des DEA theoretisch nicht so mächtig ist wie das des Von-Neumann-Modells, können in der Praxis beliebige Probleme, ggf. mit Zusatz im Modell, gelöst werden.

20

Das größte Problem dieser DEAs besteht darin, dass ihre Funktionen mit der Anzahl der Zustände in exponentieller Weise anwachsen, den Flächenbedarf an Halbleitermaterial (insbesondere Silizium) betreffend. Aus diesem Grund wählt man gerne Modelle, die aus vielen, miteinander kommunizierenden Automaten bestehen. Ein derartiges Rechnermodell wird als ‚komplexer, kooperierender Automat‘ bezeichnet.

25

## 2.3 Darstellung Boolescher Funktionen

30

Eine *Boolesche Funktion* oder *Schaltfunktion* ist eine Abbildung  $f: B^m \rightarrow B^n$ , mit  $B=\{0,1\}$ , die in der Form  $f = (f_1, f_2, \dots, f_n)$ , also als Vektor von Funktionen  $f_k: B^m \rightarrow B$  dargestellt werden kann. Im Folgenden wird daher nur von Funktionen  $f_k$  mit einem Ausgangswert ausgegangen; diese werden allgemein als  $f$  bezeichnet.

35

Es ist weiterhin bekannt, dass  $f$  in Form einer disjunktiven oder konjunktiven Normalform darstellbar ist. Für die disjunktive Normalform bedeutet dies, dass

$f = z_1 + z_2 \dots + z_k$ , mit  $k = 1, 2, \dots, 2^m$  und '+' als OR-

5 Operator (logisches ODER) (1)

und

$z_i = y_1 * y_2 * \dots * y_n$  mit  $i = 1, 2, \dots, k$  mit '\*' als AND-Operator (logisches UND) (2)

gilt. Es werden natürlich nur solche  $z_i$  verwendet, für die  
 10 die zu beschreibende Funktion den Wert TRUE oder '1' erhält. Die Bezeichnung  $y_j$  bestimmt dabei, wie ein Inputbit  $i_k$  aus dem Inputvektor  $x = (i_1, i_2, \dots, i_k)$  werden soll. Während für die Inputbits nur Werte aus der Menge  $\{0, 1\}$  zugelassen sind, muss dies für  $y_j$  geändert werden: Diesen Werten wird eines  
 15 aus den drei Attributen  $\{0, 1, -\}$  zugewiesen. Das Attribut '1' für  $y_j$  bedeutet dabei, dass  $i_k$  unverändert genommen wird, '0' bedeutet, dass  $i_k$  invertiert gewählt werden muss (notiert als  $/i_k$ ), und '-' steht für don't care; d.h.,  $i_k$  wird nicht verwendet. Werden für  $y_j$  nur die Werte  $\{0, 1\}$  als Attribute  
 20 verwendet, spricht man von der ,kanonisch disjunktiven Normalform'.

Diese Darstellung ist deswegen bedeutend, weil dadurch bei bekannter Reihenfolge der Inputbits die Teilausdrücke  $z_i$  gemäß vorstehender Gleichung (2), auch als ,Terme' bezeichnet,  
 25 als sogenannte ,Stringterme' darstellbar sind: Bei einer Reihenfolge  $i_1, i_2, i_3$  bedeutet "111", dass  $z_1 = i_1 * i_2 * i_3$  ist, "0-1" steht für  $z_2 = /i_1 * i_3$  usw.

Bei drei Inputbits ist die Menge aller möglichen Inputvektoren  $v = \{000, 001, 010, 011, 100, 101, 110, 111\}$ . Falls beispielhaft die Funktion  $f$  nur an den Eingangsvektor  $\{001, 011, 111\}$  den Wert TRUE erhält, brauchen oder dürfen auch nur diese angegeben werden; in Form von Stringterms kann dies durch 111 und  
 35 0-1 erfolgen, dies charakterisiert vollständig die gegebene Funktion und ist isomorph zur disjunktiven Normalform  
 $f = /i_1 * i_3 + i_1 * i_2 * i_3$

## 2.4 Content-Addressable Memory (CAM)

Lese-/Schreib-Speicherelemente [RAM(Random Addressable  
5 Read/Write Memory)] werden üblicherweise zur Speicherung von  
Daten und Programmen genutzt. In diesem Fall liegt eine Ad-  
resse an dem Adressbus an, und nach Ablauf einer Baustein-  
spezifischen Wartezeit ist beim Lesevorgang das gespeicherte  
Datum am Datenbus anliegend und kann weiter verwendet werden.  
10 Der Schreibvorgang ist in entsprechender Weise arbeitend.

Aus Sicht der Instanz, die die Daten erhalten möchte (z.B.  
ein Prozessor), ist die Adresse bekannt, und der gespeicherte  
Inhalt ist vorher unbekannt. Es existieren jedoch Anwendun-  
15 gen, bei der das Verhältnis genau umgekehrt ist: Der gespei-  
cherte Inhalt ist bekannt, und das Interesse ist, an welcher  
Adresse dieser Inhalt gespeichert ist, wobei die Antwort auch  
'nicht vorhanden' sein kann. Speicherelemente, die diese Art  
der Abfrage unterstützen, werden als 'Content-Addressable Me-  
20 mories' [CAMs (Inhalts-adressierbare Speicherelemente)] be-  
zeichnet.

Speicherelemente, die als CAM bezeichnet werden und dieses  
Verhalten direkt unterstützen, gelten als spezielle Bausteine  
25 und sind keineswegs häufig anzufinden. Für praktische Anwen-  
dungen kann man jedoch die Funktionalität des CAM durch übli-  
che RAM-Bausteine emulieren. Hierzu müssen für alle Daten,  
die im CAM direkt gespeichert werden würden, bei einer Abfra-  
ge jedoch nicht der Wert, sondern die Speicheradresse erge-  
30 ben, die korrespondierenden Adressen vorher berechnet werden  
und an der RAM-Adresse, die dem Datum entspricht, gespeichert  
werden.

## 2.5 Zellulare Automaten

35 Zellulare Automaten [CAs (cellular automata)] sind eine Menge  
von endliche Automaten, die in einem Feld mit feststehender

- Topologie angeordnet sind und weitere Eigenschaften besitzen (vgl. Literaturzitate [1] und [4]). Diese Menge von FSMs ist als  $n$ -dimensionales Array (meist gilt  $n = 2$ ) angeordnet, wobei bei jedem Platz feste Koordinaten gegeben sind. Jede FSM besitzt eindeutig Nachbarn, mit denen kommuniziert werden kann. Im Fall  $n = 2$  werden meist die 4 umliegenden FSMs (in den 'Himmelsrichtungen' N, E, W, S, daher auch als 'NEWS-Nachbarschaft' bezeichnet) als Nachbarn angesehen.
- Die Kommunikation mit den Nachbarn erfolgt so, dass die Zustände der direkten Nachbarn lesbar und damit auswertbar sind. Mit jedem Zeitschritt wird der Zustand aller Zellen parallel berechnet. Sollen Daten aus weiter entfernten Zellen für die Berechnung eines neuen Zustandes genutzt werden, so müssen diese Daten schrittweise von Zelle zu Zelle transportiert werden. Damit sind klassische zellulare Automaten gut zur Berechnung von Problemstellungen mit hoher Lokalität der Daten geeignet.
- CAs gelten als universelle Rechner wie die vorher diskutierten Architekturen; sie arbeiten zudem vollkommen parallel. Soll ein Netz von CAs in eine Hardwareschaltung, z.B. einen ASIC oder auch PLD, abgebildet werden, so steigt die Anzahl der Verbindungen linear mit der Zahl der Automaten an. Die Verbindungen selbst sind je nach gewählter Topologie nur relativ kurz und fest verlegt. Der Aufwand zur Kommunikation der CAs untereinander ist also relativ gering. Wird als Komplexität einer Schaltfunktion der Speicherbedarf angesehen, der nötig ist, diese Schaltfunktion in ein RAM abzubilden, so steigt die maximale Komplexität der dem Verhalten einer Zelle entsprechenden Schaltfunktion exponentiell mit der Anzahl der Eingangsvariablen und linear mit der Anzahl der Ausgangsvariablen der Funktion an. Die Zahl der Eingangsvariablen ist hier die Summe aller Bits, die nötig sind, die Zustände aller mit der Zelle verbundenen CAs einschließlich des Zustandes der Zelle selbst zu codieren. Damit ist die maximale Komplexität der Zelle im Wesentlichen durch die Anzahl der Verbin-



dungen eines jeden Automaten beschränkt.

Das Konzept der globalen zellularen Automaten [GCAs (global cellular automata)] überwindet die Einschränkungen der CAs, indem Verbindungen einer Zelle nicht nur zu ihren nächsten Nachbarn, sondern zu beliebigen Zellen im gesamten Feld erlaubt werden. Damit besitzt ein GCA keine feststehende Topologie mehr sondern ermöglicht, eine an die Problemstellung angepasste und gegebenenfalls zur Laufzeit der Berechnung sogar wechselnde Topologie zu verwenden. Dies kann zu einer erheblichen Beschleunigung in der Programmbearbeitung führen. Die Anzahl der Verbindungen eines einzelnen Automaten ist gegebenenfalls durch eine obere Grenze festgelegt. Ist nur eine einzelne Verbindung erlaubt, so spricht man von einarmigen-, im allgemeinen Fall von  $k$ -armigen GCAs.

Als Konsequenz steigt bei der Realisierung eines Feldes von GCAs der erforderliche Aufwand für die Kommunikation der Zellen untereinander mit der Zahl der Zellen stark an. Die Anzahl der möglichen Verbindungen zwischen den einzelnen Automaten steigt quadratisch mit deren Anzahl.

Die Komplexität der einzelnen Automaten selbst bzw. der zugrunde liegenden Schaltfunktion hängt wie bei den konventionellen CAs im Wesentlichen von der Anzahl der Verbindungen einer jeden Zelle ab. Soll ein GCA in eine rekonfigurierbare Schaltung (PLD) abgebildet werden, so muss jede einzelne Zelle, die ja beliebige Schaltfunktionen realisieren kann, die maximal mögliche Komplexität ermöglichen.

Werden die einzelnen Automaten auf jeweils ein Rechenwerk mit lokalem Speicher abgebildet, so kann jede Zelle auch komplexe Schaltfunktionen realisieren. Der Aufwand für eine beliebige Kommunikation aller Zellen steigt quadratisch mit der Anzahl der Zellen. Die Granularität der Schaltung wird dann bestimmt durch die Anzahl der Zellen bzw. die Bitbreite der Verbindungen zwischen den Zellen. Eine solche Schaltung kann sehr gut

GCAs realisieren, die in Anzahl der FSMs und Bitbreite den Vorgaben der Schaltung entsprechen. Es können auch komplexe Schaltfunktionen in jeder einzelnen Zelle realisiert werden. Nachteilig wirkt sich jedoch aus, dass GCAs, die in Anzahl  
5 und benötigter Bitbreite der Verbindungen nicht mit der vorgegebenen Körnigkeit übereinstimmen, nur schwierig auf die Schaltung abgebildet werden können.

Werden die einzelnen Zellen als Schaltnetz ausgeführt, so  
10 muss jede Zelle in der Lage sein, Daten von allen anderen Zellen einschließlich des eigenen Zustands zu verarbeiten. Aus diesem Grund muss jedes Schaltnetz Schaltfunktionen realisieren können, die alle binärcodierten Zustände aller Zellen als Eingabevariablen enthalten können. Die Anzahl der  
15 Ausgabevariablen der Schaltfunktion muss es lediglich ermöglichen, alle Zustände einer einzelnen Zelle binär zu codieren. Nachteilig ist hier, dass die Komplexität der Schaltfunktion exponentiell mit der Anzahl der Eingabevariablen ansteigt. Ebenfalls nachteilig ist der polynomial ansteigende  
20 Aufwand für die Kommunikation der Zellen untereinander.

Eine (re-)konfigurierbare Architektur (PLD), die zur Aufnahme eines GCA geeignet ist, muss also pro FSM eine beliebige Komplexität aufnehmen können. Dies bedeutet, dass - falls die  
25 FSMs direkt in eine Zelle des PLDs abgebildet werden sollen - die Zellen jede beliebige Funktion aufnehmen müssen. Es ist bekannt, dass sich hieraus ein exponentielles Wachstum der Zellengröße ergibt. Das konfigurierbare Netzwerk in dem PLD muss zudem vollständig ausgeführt sein, d.h., jeder Zellen-  
30 ausgang muss mit jeder anderen Zelle verbindbar sein. Das Netzwerk wächst damit quadratisch an, die Konfiguration des Netzwerks ebenfalls.

Derzeit sind keine PLD-Architekturen am Markt oder bekannt,  
35 die beide Forderungen erfüllen: Große Zellen mit einem vollständigen Designraum existieren nicht, es gibt nur kleine Zellen mit vollständigem Designraum [sogenannte ,Look-Up-

Table-Struktur` (LUT)] oder große Zellen mit unvollständigen Möglichkeiten zur Abbildung beliebiger Funktionen. Eine vollständige Verbindbarkeit aller Zellen ist nur bei großen Zellen möglich. Die Abbildung von GCAs auf existierende PLDs ist  
5 damit schwierig, vielfach sogar unmöglich.

### 3 Aufgabenstellung

Aufgabe der vorliegenden Erfindung ist es, die in der Einleitung 1 angegebene Architektur dahingehend auszugestalten, dass die vorgenannten Probleme zumindest gemindert sind. Dabei soll insbesondere die Möglichkeit aufgezeigt werden, die als ausgezeichnetes Modell zu erachtenden CGAs, die die im Algorithmus steckende Parallelität tatsächlich nutzen können,  
15 auf eine programmierbare Hardware wie ein PLD abbilden zu können. D.h., es soll eine PLD-Architektur angegeben werden, die eine Aufnahme eines beliebigen CGA ermöglicht.

### 4 Lösungsansatz

Die genannte Aufgabe wird erfindungsgemäß mit den in Anspruch 1 angegebenen Maßnahmen gelöst. Demgemäß soll die Architektur mit den eingangs genannten Merkmalen wenigstens ein einzelnes konfigurierbares und/oder rekonfigurierbares Schaltwerk umfassen, das eine Stufe mit Eingängen und eine Stufe mit Ausgängen aufweist, wobei Ausgangsvariable zumindest einiger der Ausgänge zu einem Zeitpunkt  $t_{n-1}$  die Eingangsvariablen an zugeordneten Eingängen des Schaltwerks zum Zeitpunkt  $t_n$  bilden und wobei Mittel zum Speichern der Ausgangsvariablen des  
25 Schaltwerks zwischen den Zeitpunkten  $t_{n-1}$  und  $t_n$  vorgesehen sind. Dabei brauchen nicht alle Eingänge von (gespeicherten) Ausgängen belegt zu sein; sondern es können auch freie Eingänge existieren. Außerdem können auch Ausgänge vorhanden sein, die nicht durchgekoppelt sind, also beispielsweise nur  
30 ein Endergebnis darstellen, das nicht wieder verwendet wird.

Mit den Zeitpunkten  $t_{n-1}$  und  $t_n$  sind direkt aufeinander fol-

gende Auswertungen der Schaltfunktion bezeichnet. In einer technisch günstigen Realisierung können diese Zeitpunkte von einem eingepprägten Takt mit einer Taktdauer  $T$  abgeleitet werden, so dass  $t_n - t_{n-1} = k \cdot T$  mit  $k=1,2,3,\dots$  gilt. Für eine  
5 gegebene Anwendung ist hierbei  $k$  konstant, für verschiedene Anwendungen kann es unterschiedlich gewählt werden.

Nachdem feststeht, dass die Komplexität der Schaltfunktion exponentiell mit der Anzahl der Eingabevariablen und linear  
10 mit der Anzahl der Ausgabevariablen bzw. der Anzahl der Zellen im Feld der abzubildenden GCAs steigt und der Aufwand für die Kommunikation der GCAs untereinander zumindest quadratisch mit der Anzahl der Zellen ansteigt, ist die erfindungsgemäße (re-)konfigurierbare Architektur zur Aufnahme von GCAs  
15 geeignet. Hierzu besteht sie aus wenigstens einem einzelnen konfigurierbaren Schaltwerk, dessen Ausgangsvariablen zum Zeitpunkt  $t_{n-1}$  die Eingangsvariablen des Schaltwerks zum Zeitpunkt  $t_n$  bilden. Zwischen den Zeitpunkten  $t_{n-1}$  und  $t_n$  werden die Ausgänge des Schaltwerks in einem Speicher wie insbesondere in Registern gespeichert, so dass ein komplexes Schaltwerk bzw. eine FSM entsteht. Dabei stellen die einzelnen Stufen in der Gliederung ebenfalls Schaltwerke dar; nur die Konfiguration liegt in einzelnen Schaltnetzen einer jeden Stufe, wobei jede Stufe aus einem konfigurierbaren Schaltnetz und  
20 Speicher-elementen besteht.  
25

Bei der erfindungsgemäßen Architektur sind demnach Schaltnetze als Boolesche Schaltfunktionseinheiten vorhanden, deren jeweiliger Ausgang von aktuellen Werten am Eingang der Architektur abhängt, während unter dem wenigstens einen Schaltwerk  
30 eine Boolesche Schaltfunktionseinheit zu verstehen ist, deren Ausgang von aktuellen Werten am Eingang zu dieser und endlich vielen vorangegangenen Zeitpunkten abhängt. Schaltwerke werden demnach aus Schaltnetzen und zumeist taktflankengesteuerten Speicher-elementen zusammengesetzt.  
35

Vorteilhafte Ausgestaltungen der erfindungsgemäßen Architektur gehen aus den von Anspruch 1 abhängigen Ansprüchen hervor. Dabei kann die Ausführungsform nach Anspruch 1 mit den Merkmalen eines der Unteransprüche oder vorzugsweise auch denen aus mehreren Unteransprüchen kombiniert werden. Demgemäß  
5 kann die Architektur noch folgendermaßen ausgebildet sein:

- Als Speichermittel können Registerspeicherelemente vorgesehen sein, wobei es sich hier im Allgemeinen um taktflankengesteuerte Flipflop-Elemente handelt.
- 10 - Bevorzugt ist das Schaltwerk dreistufig ausgebildet, wobei die drei Stufen hintereinander geschaltet sind. Dabei dient die zwischen der Stufe mit den Eingängen und der Stufe mit den Ausgängen angeordnete zweite Stufe mit Speicherelementen zur Verarbeitung von in der ersten Stufe erzeugten Daten und zur Weiterleitung von verarbeiteten Daten  
15 an die dritte Stufe.
- Dabei kann vorteilhaft eine erste Stufe mehrere parallelgeschaltete Speicherelemente enthalten, die über Eingabeleitungen adressierbar sind, wobei jedem Speicherelement  
20 eine Teilmenge der in einem zugehörenden, ermittelten Implikanten gebundenen Eingabeveriablen zuzuführen sind, der ersten Stufe eine zweite Stufe mit Speicherelementen nachgeordnet sein, die durch Kennungen der einzelnen Implikanten zu adressieren sind,  
25 und  
der zweiten Stufe eine dritte Stufe mit Mitteln zu einer disjunktiven Verknüpfung der Ausgabewerte der einzelnen Implikanten aus den Speicherelementen der zweiten Stufe nachgeordnet sein.
- 30 - Dabei lassen sich die einzelnen Implikanten bevorzugt durch Minimierungsverfahren ermitteln.
- Ferner kann die erste Stufe mit der zweiten Stufe über wenigstens einen Crossbar-Switch miteinander verknüpft sein.
- Als Speicherelemente können vorteilhaft CAMs und/oder RAMs  
35 vorgesehen sein.
- Besonders vorteilhaft ist wenigstens ein CGA zu integrieren.

- Als Speicherelemente können magnetoresistive Bauelemente, insbesondere vom TMR-Typ, vorgesehen sein. Entsprechende tunnelmagnetoresistive Elemente sind an sich bekannt.

5 Dabei sei unter einem Speicherelement der Architektur jedes interne Element verstanden, das im Allgemeinen aus wenigen Transistoren oder vergleichbaren Bausteinen sowie aus den eigentlichen, die Speicherfunktion ermöglichenden Elementen wie z.B. den erwähnten magnetoresistiven Elementen (pro Spei-  
10 chereinheit) zusammengesetzt ist.

Den vorstehend angegebenen weiteren Ausgestaltungen der erfindungsgemäßen Architektur liegen insbesondere die nachfolgend dargelegten Überlegungen zu Grunde:

15 Die Komplexität einer Schaltfunktion in dem gewählten Schaltwerk steigt zwar linear mit der Anzahl der Ausgabevariablen an, jedoch entfällt sämtlicher Aufwand für die Kommunikation der Zellen untereinander. In eine solche Schaltfunktion können viele einzelne Automaten mit wenigen Ausgabevariablen ab-  
20 gebildet werden, oder wenige GCAs mit vielen Ausgabevariablen oder auch eine Mischung verschiedener Zellen. Damit ist keine Granularität vorgegeben, und die Kommunikation der FSMs untereinander ist prinzipiell vollständig möglich. Eine Grenze  
25 ist jedoch durch die maximale Komplexität der Schaltfunktion gesetzt, die das Schaltnetz aufnehmen kann.

Die Benutzung eines Schaltwerks mit einer großen Anzahl von Eingängen – dies wird im allgemeinen Fall entstehen, wenn ein  
30 GCA mit einer Anzahl von FSMs abgebildet wird – bedeutet, dass wiederum eine Art exponentielle Abhängigkeit des Flächenbedarfs von der Anzahl der Eingänge entstehen kann. Als obere Grenze gilt ein Wachstum der Fläche mit  $\exp(\text{Anzahl Zustände})$ , wenn jeder Zustand in einem Bit codiert wird; im  
35 allgemeinen Fall wird der Flächenbedarf geringer sein. Da eine universelle Schaltung jedoch den Maximalfall beinhalten muss, wäre das exponentielle Wachstumsgesetz anzuwenden.

- Hier ist es als besonders vorteilhaft anzusehen, die Schaltung, die das Schaltwerk aufnimmt, in drei spezielle Abschnitte/Stufen einzuteilen. Dazu wird nachstehend eine
- 5 (re)konfigurierbare Schaltung dargestellt, die Schaltfunktionen mit einer großen Zahl von Eingangsvariablen und einer großen Zahl von Ausgangsvariablen als Schaltwerk realisieren kann.
- 10 Um ein rekonfigurierbares Schaltwerk für Schaltfunktionen zu entwerfen, werden als Ausgangsüberlegung zwei Möglichkeiten betrachtet:
- 15 Zum einen ist es möglich, eine Schaltfunktion komplett in einem RAM-Speicher abzulegen. Die Eingabevariablen der Schaltfunktion bilden die Adressbits und adressieren für jede mögliche Kombination von Eingabevariablen eine Speicherzelle. Der Inhalt dieser Speicherzelle entspricht dann dem Wert der Schaltfunktion, die Datenleitungen des
- 20 Speicherelements bilden die Ausgabevariablen der Funktion. Der Vorteil dieses Konzepts liegt im einfachen Aufbau der Schaltung, der einfachen Rekonfigurierbarkeit, der hohen Integrationsdichte von Speicherelementen und der festen Zeitdauer, die die Auswertung der Schaltfunktion
- 25 benötigt. Allerdings steigt die Anzahl der benötigten Speicherzellen, also die Größe des benötigten Speicherelementes exponentiell mit der Anzahl der Eingabevariablen an. Aus diesem Grunde können nur kleine Schaltfunktionen auf diese Weise abgebildet werden.
- 30 Dies ist Stand der Technik in PLDs und wird als Look-Up-Table-Struktur bei FPGAs eingesetzt, meist mit 4 binärwertigen Eingangsvariablen und 1 binärwertigen Ausgang.
- 35 Eine zweite Möglichkeit, Schaltfunktionen in ein Schaltnetz abzubilden, besteht darin, Gatter in einem 2- oder mehrstufigen Netz konfigurierbar anzuordnen. Damit ist es

möglich, Schaltfunktion mit einem minimalen Verbrauch von Gattern in Schaltnetze abzubilden. Die Schaltkreistheorie stellt hier günstige Darstellungsformen von Schaltfunktionen wie z.B. die Reed-Muller-Form, oder auch leistungsfähige Algorithmen zur Logikminimierung bereit. Der Vorteil dieses Ansatzes besteht im minimalen Verbrauch von Gattern und in der Möglichkeit, leistungsfähige Verfahren und Algorithmen zur Minimierung zu nutzen (vgl. Literaturzitate [2] und [3]). Dieses Verfahren kann gut zur Darstellung einer festen Schaltfunktion z.B. der Realisierung eines ROMs genutzt werden. Über Hardwarebausteine wie z.B. Crossbar-Switches können die einzelnen Gatter rekonfigurierbar verschaltet werden, jedoch steigt hier der Aufwand für die Rekonfigurierbarkeit exponentiell mit der Anzahl der Eingabevariablen der Schaltfunktion an.

5 Erläuterung der Erfindung anhand eines konkreten Ausführungsbeispiels

20 Um die Vorteile der ersten Möglichkeit, die hohe Integrierbarkeit, die Rekonfigurierbarkeit und die Einfachheit der Schaltung, und die Vorteile der 2. Variante, den geringen Verbrauch von Gatterfunktionen und die Anwendbarkeit moderner Verfahren, verbinden zu können, ist erfindungsgemäß folgender  
25 Ansatz vorgesehen:

Die Grundidee der angenommenen Ausführungsform ist die Entwicklung eines rekonfigurierbaren Schaltwerks, welches soviel logische Funktionalität wie möglich in RAM-Bausteinen darstellt, um den Vorteil der hohen Integrationsdichte zu nutzen. Diese RAM-Bausteine sollen in einer mehrstufigen Schaltung rekonfigurierbar miteinander verbunden werden, um nicht komplette Schaltfunktionen speichern zu müssen.

35 1. Stufe:

Die Eingabevariablen der Schaltfunktion werden durch die Eingabeleitungen des Schaltnetzes repräsentiert. Die erste Stufe



der Schaltung besteht aus mehreren parallel geschalteten Speicherelementen, die durch die Eingabeleitungen adressiert werden. Jedem Speicherelement ist dabei eine Teilmenge der Eingabeleitungen und damit der Eingabevariablen der Schaltfunktion zugeordnet. Die durch Minimierungsverfahren ermittelten Implikanten einer Schaltfunktion (minimierte Zi aus Gl. (2) bzw. deren Stringterm-Darstellung) werden in Speicherelementen der Eingangsstufe abgespeichert. Dazu werden in jedem Speicherelement, dem im Implikanten gebundene Eingabevariablen zugeordnet sind, die Belegung der Eingabevariablen im Implikanten und eine eindeutige Kennung zu diesem Implikanten abgespeichert. Da jedem Baustein nur ein Teil der Eingabevariablen zugeordnet sind, wird auch nur jeweils ein Teilimplikant gespeichert. In Speicherelementen, denen keine im Implikanten gebundenen Variablen zugeordnet sind, wird entsprechend kein Teil des Implikanten gespeichert. Liegt an den Eingabeleitungen des Schaltnetzes eine Bitkombination an, so geben alle Speicherelemente, die zur Bitkombination passende Teilimplikanten enthalten, deren Kennungen über die Datenleitungen an die 2. Stufe der Schaltung weiter.

## 2. Stufe:

Die Kennungen der (Teil-)Implikanten adressieren einen Speicher in einer zweiten Stufe. In diesem Speicher sind die zum jeweiligen Implikanten gehörigen Bitmuster der Kennungen und die Ausgabewerte der Schaltfunktion gespeichert. Entspricht das Bitmuster an Kennungen, die von der ersten Stufe geliefert werden, dem eines gespeicherten Implikanten, so liegt dieser Implikant an den Eingangsleitungen der Schaltung an. Die 2. Stufe der Schaltung leitet dann die Ausgabewerte aller Implikanten, die an den Eingabeleitungen anliegen, über die Datenleitungen an die 3. Stufe weiter.

## 3. Stufe:

In der 3. Stufe werden die Ausgabewerte der einzelnen Implikanten disjunktiv (OR) verknüpft und bilden so das Ergebnis der Schaltfunktion.

Die Erfindung wird nachfolgend unter Bezugnahme auf die Zeichnung anhand eines konkreten Ausführungsbeispieles unter Berücksichtigung der vorstehenden Überlegungen noch weiter erläutert. Die Zeichnung umfasst folgende Teile, wobei deren Figuren 1 bis 3 zum Stand der Technik unter Abschnitt 2 bereits angesprochen wurden:

- 10 Figur 1 zeigt den prinzipiellen Aufbau einer Von-Neumann-Architektur gemäß dem Stand der Technik,  
Figur 2 zeigt die generelle Struktur eines PLD gemäß dem Stand der Technik,  
Figur 3 zeigt den prinzipiellen Aufbau einer FSM in Form eines Mealy-Automaten gemäß dem Stand der Technik,  
15 Figur 4 zeigt ein Ausführungsbeispiel eines Speicheraufbaus einer erfindungsgemäßen Architektur,  
Figur 5 zeigt die Abbildung von Springtermen auf RAM, wobei Teilfigur a) partielle Springterme, Teilfigur b) die  
20 Abbildung auf ein Tag-RAM und Teilfigur c) die Abbildung auf ein konventionelles RAM veranschaulichen,  
Figur 6 zeigt die Abbildung des Ergebnisses der Stufe 1 einer erfindungsgemäßen Architektur auf eine RAM-Kombination in Stufe 2,  
25 Figur 7 zeigt eine endgültige Architektur für das Beispiel  $[(0,1)^{12} \rightarrow (0,1)^{12}$ -Funktion],  
Figur 8 zeigt eine erfindungsgemäße Architektur für ein Schaltwerk mit großem Schaltnetz zur Aufnahme eines GCA  
30 und  
Figur 9 eine erfindungsgemäße rekonfigurierbare Architektur zur Aufnahme von CGAs.

35 Dabei sind in den Figuren sich entsprechende Teile jeweils mit denselben Bezugszeichen versehen.

Für das Ausführungsbeispiel gemäß den Figuren 4 bis 9 sei ei-

ne erfindungsgemäße Architektur mit drei Stufen, wie vorstehend angesprochen, angesetzt, und zwar für eine Schaltfunktion mit 12 Eingabevariablen, 10 Implikanten und 8 Ausgabevariablen: Tabelle 1 zeigt hierfür alle Implikanten (auch als  
 5 ,Minterme' bezeichnet) für eine Beispielfunktion an. Die Darstellung der Stringterme ist so gewählt, dass hierbei drei Vierergruppen entstehen.

10 Tabelle 1: Beispiel für eine Implikantentabelle, dargestellt durch Stringterme

Implikanten:

1. 0-11 ---- 1100	6. 0100 11-- 0000
2. 1101 11-- 0000	7. ---- 0001 0000
3. -0-1 0001 ----	8. ---- 0001 ----
15 4. ---- ---- --10	9. ---- ---- 0000
5. 0100 0001 ----	10. -0-1 0001 1100

Betrachtet man nun diese Tabelle spaltenweise, wird man feststellen, dass nur wenige verschiedene Kombinationen in den  
 20 Stringtermen vorkommen. Bei einer zweiwertigen Darstellung könnte es für jede Spalte hier 24 = 16 verschiedene Kombinationen geben, bei dreiwertiger entsprechend 34 = 81. In dem Beispiel kommen hiervon nur 5, 3 und 4 für die Spalten 1 - 3 vor, wobei eine Eingangskombination jeweils komplett '-' ist.

25 Liegt am Eingang eine Bitkombination als Parameter der Schaltfunktion an, so liefert ein Speicherelement, welches einen Teilimplikanten mit der anliegenden Bitkombination speichert, die Kennung des zugehörigen Implikanten zurück.  
 30 Dieses Speicherelement der ersten Stufe ist in Figur 4 als 3-wertiges CAM ausgeführt, d.h., die Eingangsvektoren, die real als zweiwertige Information an dem Adressbus anliegt, wird mit gespeicherter dreiwertigen Informationen verglichen. Als Ausgabe wird eine zu dem Treffer gespeicherte Kennung, eben-  
 35 falls dreiwertig, ausgegeben.

Alle Kennungen zusammen bilden die Kennung der Implikanten-

kombination, die am Eingang des Schaltnetzes anliegt. Diese Implikantenkombination kann dabei durchaus mehrere Implikanten umfassen. So können im dargestellten Beispiel z.B. die Implikanten 3, 4 und 8 oder die Kombination der Implikanten 4, 5 und 8 anliegen. Im ersten Fall liegt die Bitkombination 100001 an der 2. Stufe an, im zweiten Fall die Kombination 110001.

Die Kennung der Implikantenkombinationen wird in der 2. Stufe der Schaltung erkannt und liefert für jeden beteiligten Implikanten den zugehörigen Ausgabewert der Schaltfunktion. Diese zweite Stufe besteht nun aus einem dreiwertigen RAM, d.h., am die Adressbusinformationen dieser Stufe sind dreiwertig, die gespeicherten Daten allerdings zweiwertig.

In der 3. Stufe der Schaltung werden die Ausgabewerte der anliegenden Implikanten disjunktiv verknüpft und bilden zusammen den Funktionswert der Schaltfunktion.

## 5.1 Speicherelemente Stufe 1

Als Speicherelement zum Aufnehmen der Teilimplikanten kann - wie schon erwähnt - ein sehr spezieller Baustein bzw. eine sehr spezielle Architektur zum Einsatz kommen, hier mit dreiwertigem CAM gemäß Figur 4 bezeichnet. Hierbei muss prinzipiell noch das Problem der Mehrfachübereinstimmung diskutiert werden, was aber im Zusammenhang mit anderen Realisierungsmöglichkeiten erfolgen soll.

Möglich als Einsatz für die Stufe 1 ist auch ein vollassoziativer Cache. Hier können die Teilimplikanten als sogenannter Tag gespeichert werden, das gecachte Datum dient als Kennung des erkannten Implikanten. Enthält jedoch ein Teilimplikant ungebundene Variablen, die beim Vergleich mit anliegenden Bitkombinationen als Don't-Care(DC)-Stellen zum Ausdruck kommen, so muss für alle Belegungen dieses Implikanten, die den Vergleich mit DC erfüllen, ein Tag im Tag-RAM angelegt wer-

den. Weiterhin ergeben sich durch den Vergleich mit DC  
Überschneidungen von Teilimplikanten. So gehört z.B. die Bit-  
kombination 0011 im ersten Teilimplikantenspeicher des obigen  
Beispiels sowohl zum Implikanten 1 als auch zum Implikanten  
5 3. Es sind also nicht nur Kombinationen von Implikanten mög-  
lich, sondern auch Kombinationen von Teilimplikanten.

Aus diesem Grunde werden zu einer Realisierung als Teilimpli-  
kantenspeicher normale, d.h. zweiwertige RAM-Bausteine bzw. -  
10 Architekturen verwendet. Jedes dieser RAMs wird durch einen  
Teil der Eingangsleitungen des Schaltnetzes adressiert. An  
den Adressen, deren Bitkombination jeweils demselben Teil-  
implikanten mit DC-Stellen entsprechen (jede DC-Stelle in  
einem Stringterm bedeutet, dass die Anzahl der zutreffenden  
15 Stellen bei binärer Codierung um den Faktor 2 erhöht wird),  
wird jeweils die gleiche Kennung im Speicher abgelegt. Ist  
die Ausgangsbreite des verwendeten RAMs größer als die zur  
Darstellung der Kennungen notwendige Bitbreite, so können die  
weiteren Bits als Kontextwert genutzt werden. Dieser Kontext  
20 kann z.B. eine ungültige Belegung der Eingangsvariablen der  
Schaltfunktion anzeigen.

Da ein normales RAM keine Statusanzeige für ein nicht vorhan-  
denes Datum wie z.B. ein Tag-RAM mit seinem Cache-Miss-  
25 Ausgang besitzt, muss eine Bitkombination auf denjenigen Da-  
tenleitungen, die an die 2. Stufe zum Vergleich der Bitkombi-  
nationen verwendet werden, als Kennzeichnung für keinen an-  
liegenden Teilimplikanten verwendet werden. In Figur 5 c) ist  
dies durch die Kennung 8 gegeben.

30

## 5.2 Speicherelemente Stufe 2

Ebenso wie im bei den RAMs der ersten Stufe zum Speichern der  
Teilimplikanten müssen auch beim Vergleich der Implikanten-  
35 kombinationen in der 2. Stufe der Schaltung DCs berücksich-  
tigt werden. Deshalb wird auch hier ein normales RAM verwen-  
det. Dieses RAM wird mit der Kennung der Implikantenkombina-

tion adressiert.

Da wieder mehrere Adressen derselben Implikantenkombination entsprechen können, muss der Speicher der zweiten Stufe auf-  
5 geteilt werden: die Bitkombinationen der ersten Stufe adressieren ein RAM der zweiten Stufe. Dort ist für jede gültige Implikantenkombination ein Index abgelegt, der wiederum ein RAM adressiert, welches seinerseits die Ausgabevariablen der beteiligten Implikanten enthält. So können die verschiedenen  
10 Adressen, die sich durch die Implikantenkombination mit Don't-Care-Stellen ergeben, auf denselben Index der Ausgabewerte der Schaltfunktion abgebildet werden.

Figur 6 zeigt eine entsprechende Abbildung des Ergebnisses  
15 der Stufe 1 auf eine RAM-Kombination in Stufe 2.

### 5.3 Endgültige Architektur gemäß Figur 7

Da ein RAM nur einen einzigen Index liefern kann, müssen in  
20 einer 1. Stufe eines Schaltwerks im Speicher der Ausgabewerte die disjunktiv verknüpften Ausgabewerte aller an der erkannten Kombination beteiligten Implikanten aufgenommen werden. Damit muss das Ausgabe-RAM in dieser Stufe alle möglichen Funktionswerte der Schaltfunktion speichern können. Da die  
25 Anzahl der möglichen Funktionswerte exponentiell mit der Anzahl der Ausgabevariablen einer Schaltfunktion ansteigt, werden mehrere Kombinations- und Ausgabe-Speicher der 2. Stufe des Schaltwerks parallel verwendet und deren Ausgabekombinationen disjunktiv verknüpft. Damit können alle Funktionswerte  
30 einer Schaltfunktion erzeugt werden.

Um die Kapazität der einzelnen Kombinationsspeicher besser ausnutzen zu können, werden alle Datenleitungen der Implikantenspeicher mit allen Adressleitungen der Kombinationsspeicher  
35 über einen Crossbar-Switch verbunden. Damit können beliebige Datenleitungen die Adressierung der Kombinations-RAMs übernehmen. Nicht verwendete Datenleitungen können über den

Crossbar-Switch als Kontext-Information weitergeleitet werden.

In einem letzten Schritt wird in einer 3. Stufe des Schaltwerks der erzeugte Ausgabewert bitweise über die Exklusiv-Oder-Funktion mit einem Registerinhalt verknüpft, um einzelne Ausgabevariablen invertieren zu können und so ggf. kleinere Logikminimierungen zu erhalten. Damit besteht die 3. Stufe der Schaltung aus der disjunktiven Verknüpfung der Ausgabekombinationen und der anschließenden möglichen Invertierung einzelner Ausgabebits.

Insgesamt ergibt sich das Prinzipschaltbild der Figur 7 für ein erfindungsgemäßes Schaltwerk.

15

#### 5.4 Diskussion der Architektur

Die Intention der Erfindung ist es, sowohl eine RAM-basierte Architektur zur Implementierung großer Schaltnetze als auch - gewissermaßen als Anwendung zur Aufnahme einer universellen Maschine - diese Architektur zur Aufnahme von GCAs anzubieten. Zur Aufnahme einer beliebigen Funktion im RAM muss der Speicherplatz in dem Speicher exponentiell mit der Anzahl der Eingänge (und linear mit der Anzahl der Ausgänge) wachsen. Im Fall des obigen Beispiels bedeutet dies, dass eine beliebige Funktion mit 12 Ein- und 12 Ausgängen einen Speicherbedarf von  $4096 * 12$  bit entsprechend 6144 Bytes hätte. Bei 32 Eingängen und 8 Ausgängen wären dies bereits 4 GByte an Speicherkapazität.

30

Die vorgeschlagene Architektur eines mehrstufigen Netzwerks beinhaltet lediglich 211,5 Bytes RAM, nämlich:

35	3x Implikanten-RAM 16x4	24 Bytes
	3x Kombinations-RAM 64x4	96 Bytes
	3x Ausgabe-RAM 16x12	72 Bytes
	Crossbar-Switch-Konfiguration	18 Bytes

<u>Invertierung 12x1</u>	<u>1,5 Bytes</u>
Summe	211,5 Bytes

5 Damit liegt der wesentliche Vorteil darin, dass diese Architektur erheblich platzsparender ist als eine LUT-basierte Architektur. Hierbei ist zu berücksichtigen, dass nicht jede Funktion auf diese Weise darstellbar ist.

10 Um eine Applikation in diese Architektur abbilden zu können, müssen mehrere notwendige Bedingungen erfüllt sein. Die erste Bedingung ist diejenige, dass die Anzahl der verschiedenen Teil-Stringterme, die in einer Spalte vorhanden sind, auf die RAMs der ersten Stufe abgebildet werden können. Dies ist automatisch dadurch erfüllt, dass diese RAMs alle Kombinationen  
15 aufnehmen (weil sie CAMs emulieren), lediglich die Eingangsbreite der Schaltung muss für Applikation ausreichen.

Die zweite Bedingung schließt sich hier an: Die Anzahl der verschiedenen Teil-Stringtermkombinationen, die in der Appli-  
20 kation nach Minimierung enthalten sind, muss codierbar sein. Dies bedeutet, dass eine Anzahl von Speicherstellen zur Verfügung stehen muss. Zur Effizienzabschätzung sei  $m$  die Eingangsbreite des Schaltnetzes. Dies würde bedeuten, dass  $2m$  Speicherzellen benötigt würden, um die komplette Funktion  
25 darzustellen.

Wenn  $k$  nun die Parallelität der Bausteine (Anzahl der Bausteine) und  $2s$  die Kapazität eines Bausteins ist, so muss für eine effizientere Speicherung der Applikation die Ungleichung  
30  $k * s \leq m-1$  (3) gelten. Je deutlicher die Unterschreitung ausfällt, desto effizienter war die Implementierung.

35 Bedingung 3 bedeutet, dass die Ausgangsbreite geeignet gewählt sein muss.

6 Abbildung von GCAs auf die Architektur



Zur Abbildung von GCAs auf die erfindungsgemäße Architektur müssen noch Speicherelemente eingeführt werden, die takt-gesteuert das Fortschreiten in der Rechnung speichern. Dies hat seine Ursache darin, dass GCAs als Array von FSMs definiert sind, und diese sind in der Regel synchronisiert. Hier ist angenommen, dass ein globaler Takt zur Synchronisation genommen wird. Alle Implementierungen von nicht-globalen, insbesondere nicht in gegenseitiger Beziehung stehenden Takten würden zu wesentlichen Problemen führen, sind jedoch in der Praxis selten anzutreffen.

Figur 8 zeigt ein weiteres Beispiel für eine konfigurierbare Architektur der vorgeschlagenen Art, nunmehr ausgestattet mit Registern zur Speicherung von Zuständen. Zusätzlich ist ein weiterer Crossbar-Switch eingefügt, der an dieser Stelle u.a. dazu dient, Ein- und Ausgabeschnittstellen für den Rechner bereitzustellen. Dies ist zwar für das grundlegende Verständnis der Architektur unerheblich, im praktischen Betrieb jedoch notwendig, weil ein Rechner mit Außenanschlüssen versehen sein muss.

Der Speicherbedarf dieser beispielhaften Architektur berechnet sich zu

25	8x minterm-RAM 256x8	2 KBytes
	8x combination-RAM 64Kx8	512 KBytes
	8x output-vector-RAM 256x64	16 KBytes
	2xCrossbar-Switch configuration	1 KBytes
30	<u>Inverting register 64x1</u>	<u>8 Bytes</u>
	Sum	531 KBytes

Damit ist verdeutlicht, wie gering die Speicheranforderungen sind; ein RAM mit  $264 * 64$  bit (=267 Bytes) ist jedenfalls nicht in der Herstellung möglich. Sollten bei einer Applikation auf dieser Architektur Leitungen von den RAMs der ersten Stufe ungenutzt bleiben, können diese als Kontextinformatio-

nen genutzt werden. Eine Anwendung besteht dabei in der Kontextumschaltung, die für die RAMs der zweiten Stufe zusätzlich möglich sein könnte. Werden also beispielsweise bei einem RAM nur 14 Adressbits benötigt, dann können die Informationen für diese 14 bit viermal gespeichert werden, also in vier verschiedenen Kontexten stehen.

Aus Figur 9 ist der prinzipielle Aufbau einer rekonfigurierbaren Architektur nach der Erfindung zu entnehmen, wie sie zur Aufnahme von CGAs geeignet ist. Dieser Aufbau stellt eine Verallgemeinerung des Aufbaus nach Figur 8 dar. Insbesondere sind die RAM-Stufen 1 und 2 durch gestrichelte Linien verdeutlicht.

## 7 Literaturzitate

- [1] Rolf Hoffmann, Klaus-Peter Völkman, Wolfgang Heenes:  
"Globaler Zellularautomat (GCA): Ein neues massivparalleles Berechnungsmodell", Mitteilungen - Gesellschaft für Informatik e.V., Parallel-Algorithmen und Rechnerstrukturen, ISSN 0177-0454 Nr. 18, 2001, Seiten 21-28;  
<http://www.ra.informatik.tu-darmstadt.de/publikationen/publik.html>
- [2] R.K.Brayton et.al.: "Logic Minimization Algorithms for VLSI Synthesis", Kluwer Academic Publishers, USA 1984.
- [3] Mike Trapp: "PLD-design methods migrate existing designs to high-capacity devices", EDN Access, Febr. 1994;  
<http://www.reed-electronics.com/ednmag/archives/1994/021794/04df1.htm>
- [4] Wolfgang Heenes, Rolf Hoffmann, Klaus-Peter Völkman:  
"Architekturen für den globalen Zellularautomaten".19th PARS Workshop, March 19-21, 2003 Basel;  
<http://www.ra.informatik.tu-darmstadt.de/publikationen/pars03.pdf>

## Patentansprüche

1. Rekonfigurierbare Architektur einer Rechneinrichtung mit wenigstens einem einzeln konfigurierbaren und/oder rekonfigurierbaren Schaltwerk, das eine Stufe mit Eingängen und eine Stufe mit Ausgängen aufweist, wobei Ausgangsvariable zumindest einiger der Ausgänge zu einem Zeitpunkt  $t_{n-1}$  die Eingangsvariablen an zugeordneten Eingängen des Schaltwerks zu einem Zeitpunkt  $t_n$  bilden und wobei Mittel zu einem taktgesteuerten Speichern der Ausgangsvariablen des Schaltwerks zwischen den Zeitpunkten  $t_{n-1}$  und  $t_n$  vorgesehen sind.
2. Architektur nach Anspruch 1, dadurch gekennzeichnet, dass die Speichermittel Registerspeicherelemente sind.
3. Architektur nach Anspruch 1 oder 2, gekennzeichnet durch eine dreistufige Ausbildung des Schaltwerks mit einer Hintereinanderschaltung dreier Stufen.
4. Architektur nach Anspruch 3, gekennzeichnet
- durch eine erste Stufe aus mehreren parallel geschalteten Speicherelementen, die über Eingabeleitungen adressierbar sind, wobei jedem Speicherelement eine Teilmenge der in einem zugehörenden, ermittelten Implikanten gebundenen Eingabevariablen zuzuführen sind,
  - durch eine der ersten Stufe nachgeordnete zweite Stufe mit Speicherelementen, die durch die Kennungen der einzelnen Implikanten zu adressieren sind, und
  - durch eine der zweiten Stufe nachgeordnete dritte Stufe mit Mitteln zu einer disjunktiven Verknüpfung der Ausgabe-  
werte der einzelnen Implikanten aus den Speicherelementen der zweiten Stufe.
5. Architektur nach Anspruch 4, gekennzeichnet durch eine Ermittlung der Implikanten durch Minimierungsverfahren.

6. Architektur nach Anspruch 4 oder 5, dadurch gekennzeichnet, dass die erste Stufe mit der zweiten Stufe über wenigstens einen Crossbar-Switch miteinander verknüpft ist.

5 7. Architektur nach einem der vorangehenden Ansprüche, gekennzeichnet durch CAMs- und/oder RAMs als Speicherelemente.

8. Architektur nach einem der vorangehenden Ansprüche, gekennzeichnet durch eine Integration wenigstens eines CGAs.

10

9. Architektur nach einem der vorangehenden Ansprüche, gekennzeichnet durch magnetoresistive Speicherelemente, insbesondere vom TMR-Typ.

15

1/7

FIG 1

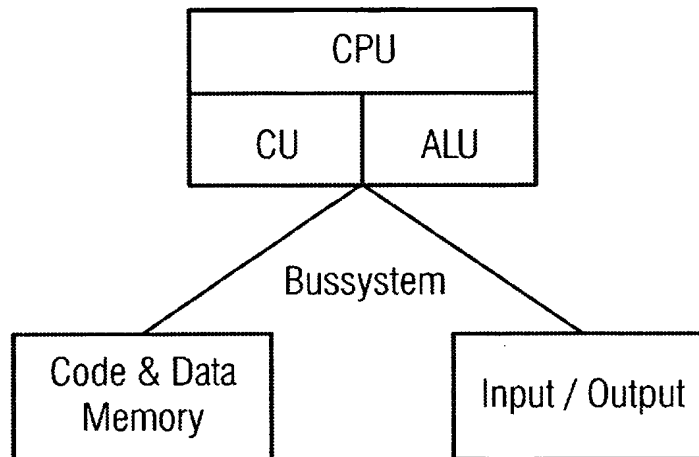
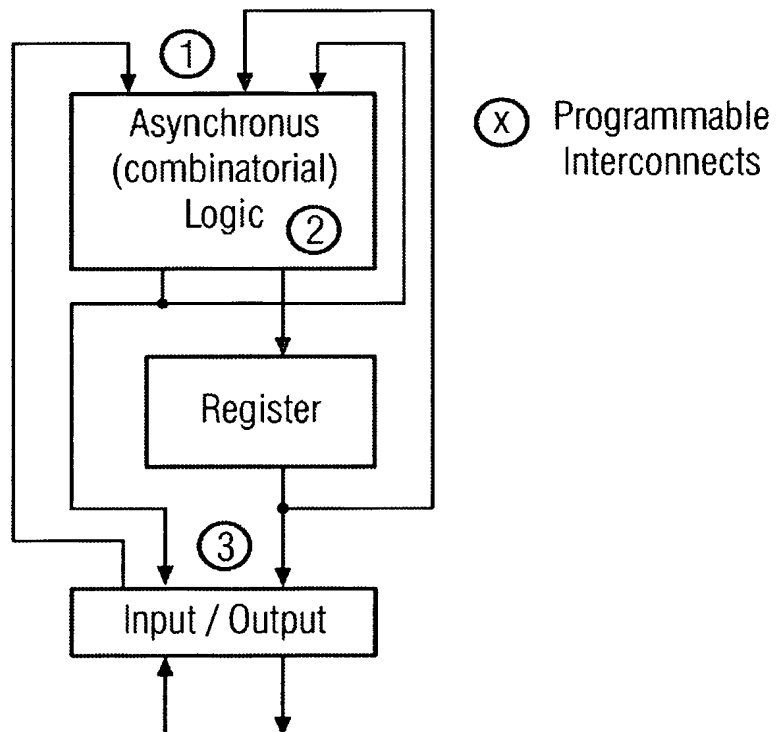


FIG 2



2/7

FIG 3

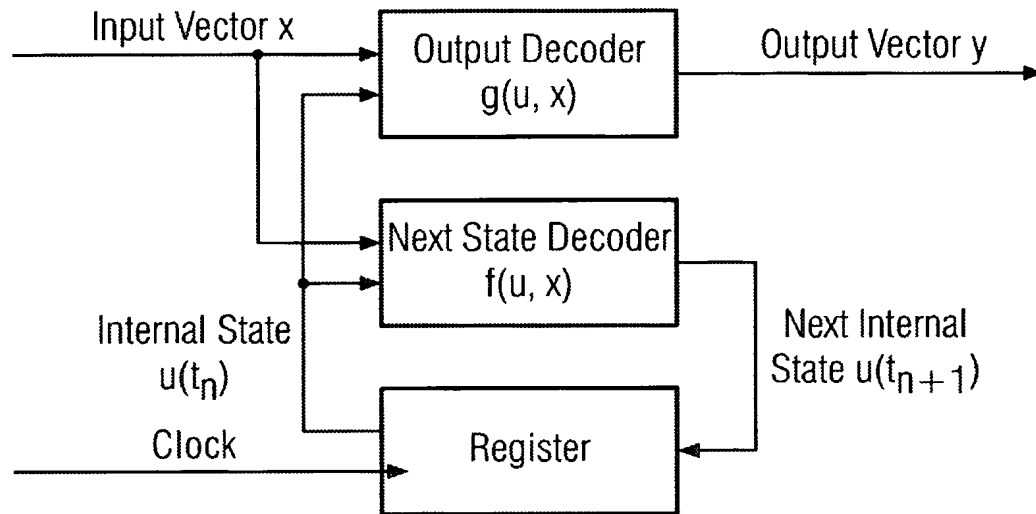


FIG 4

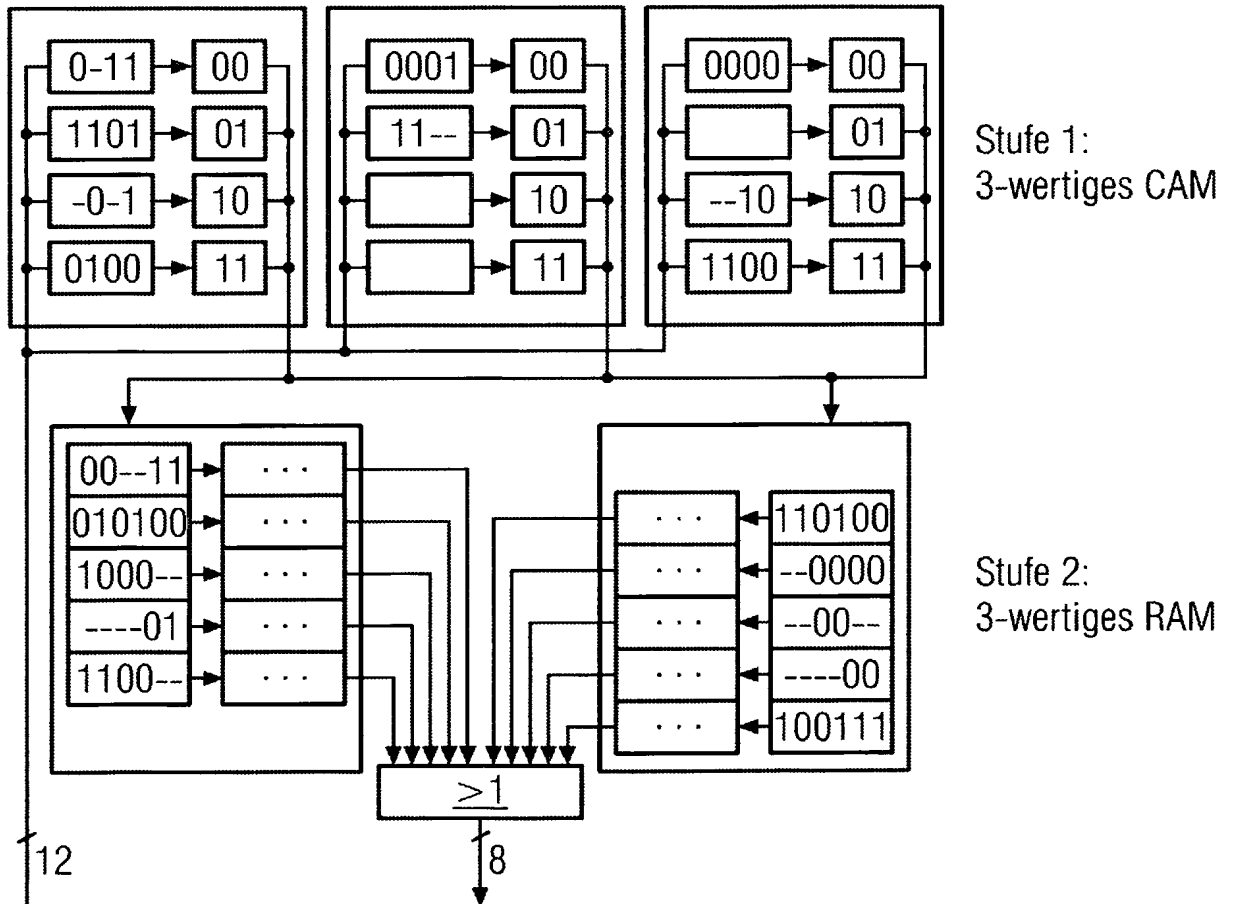


FIG 5a

partial  
Minterms  
Bits 0-3

1. 0-11
2. 1101
3. -0-1
4. ----
5. 0100
6. 0100
7. ----
8. ----
9. ----
10. -0-1

FIG 5b

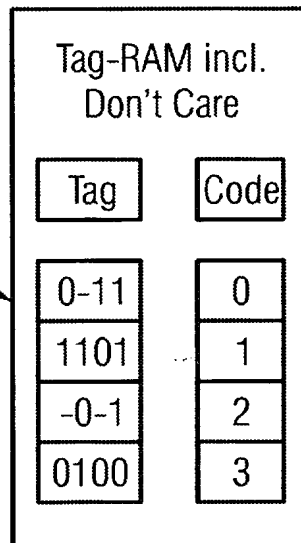
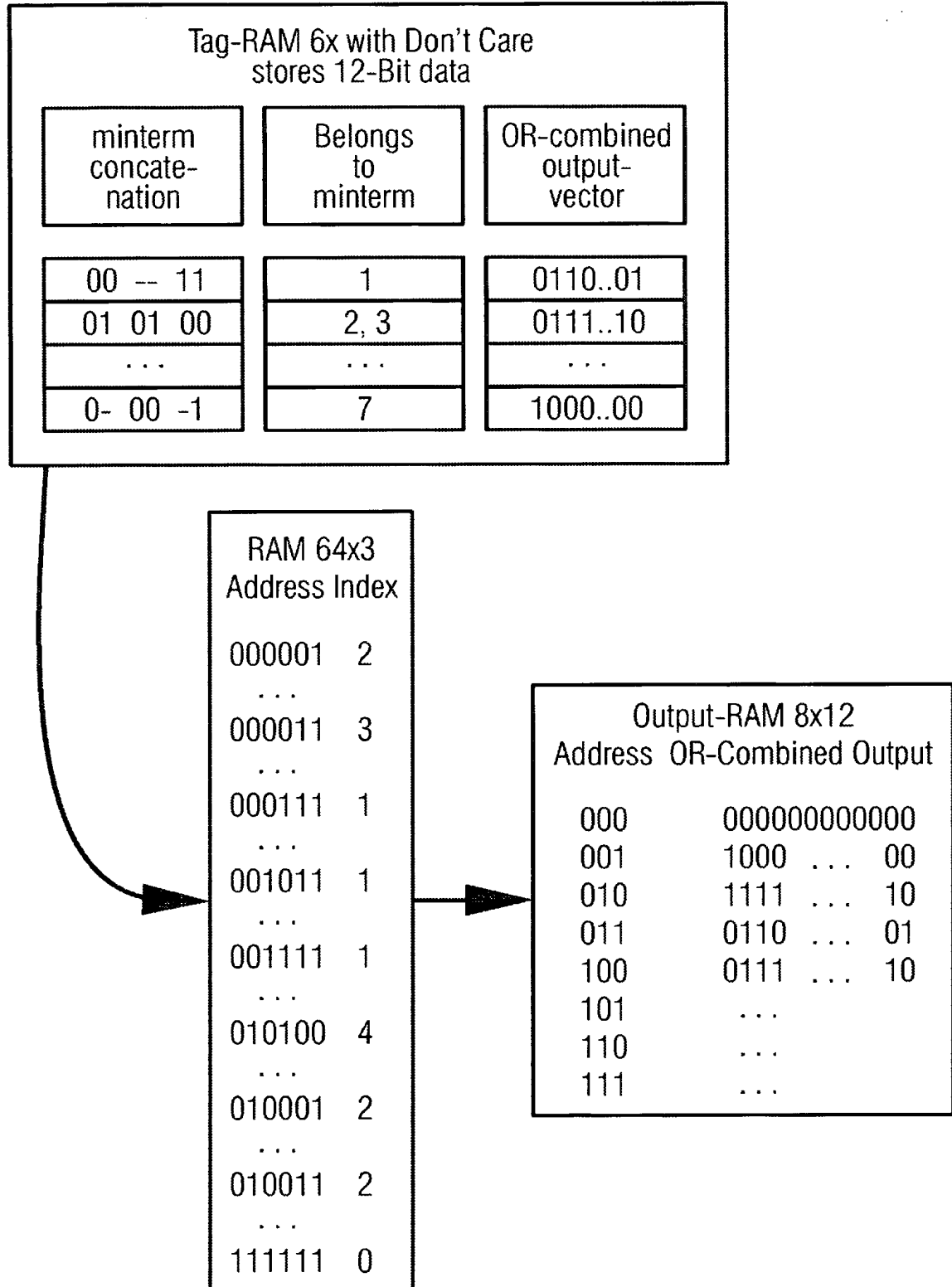


FIG 5c

RAM 16x4 Address Code	
0000	8
0001	2
0010	8
0011	4
0100	3
0101	8
0110	8
0111	0
1000	8
1001	2
1010	8
1011	2
1100	8
1101	1
1110	8

4/7

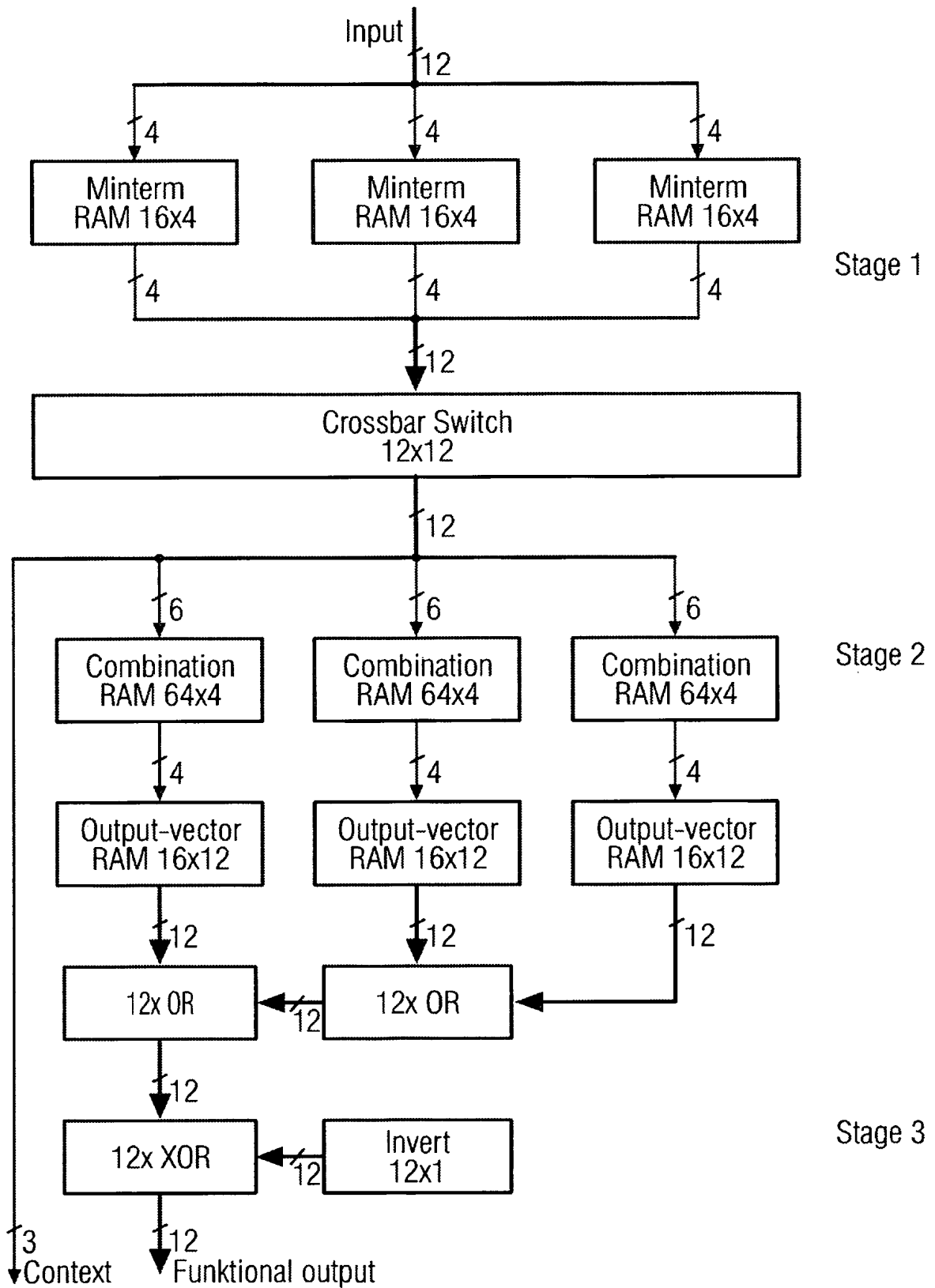
FIG 6





5/7

FIG 7



6/7

FIG 8

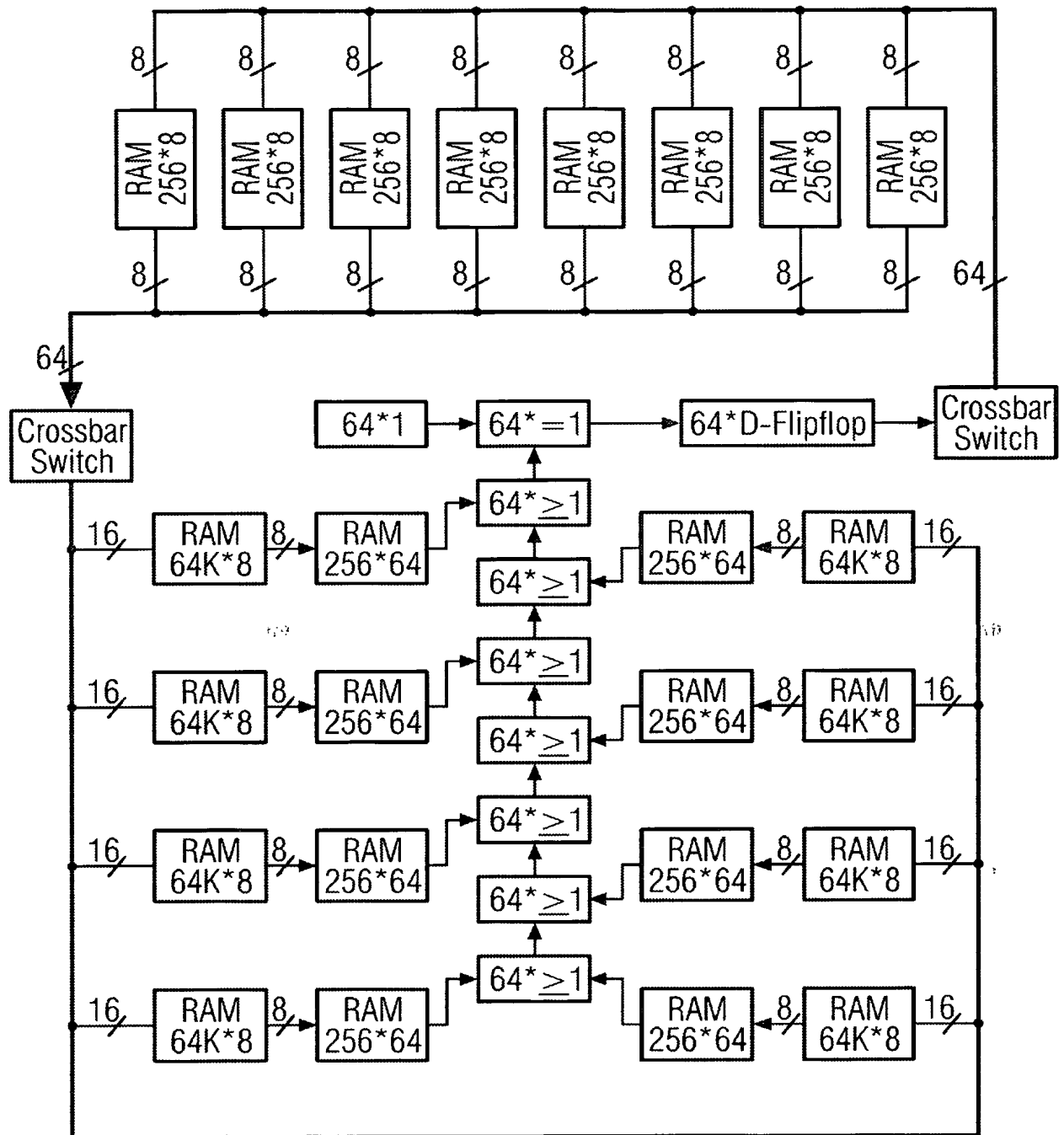
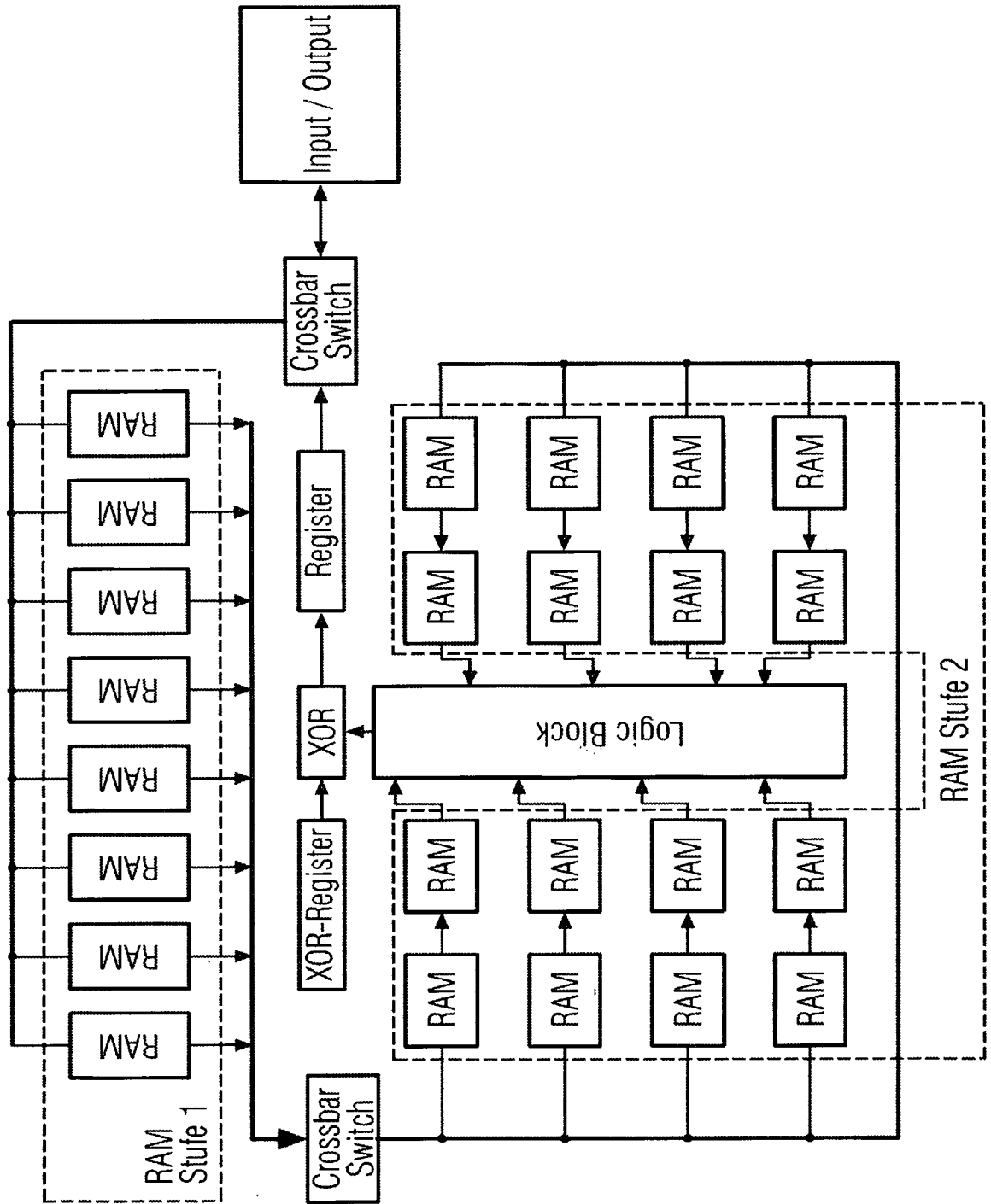


FIG 9



## INTERNATIONAL SEARCH REPORT

International Application No

PCT/EP2005/050500

**A. CLASSIFICATION OF SUBJECT MATTER**

IPC 7 G06F15/80 G06F17/50

According to International Patent Classification (IPC) or to both national classification and IPC

**B. FIELDS SEARCHED**

Minimum documentation searched (classification system followed by classification symbols)

IPC 7 G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

EPO-Internal, INSPEC

**C. DOCUMENTS CONSIDERED TO BE RELEVANT**

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	RECHENBERG, POMBERGER: "Informatik-Handbuch, 3. Auflage" 2002, HANSER, MÜNCHEN, XP002327006 page 275	1, 2, 8, 9
Y		3, 7
A		4-6



Further documents are listed in the continuation of box C.



Patent family members are listed in annex.

° Special categories of cited documents:

\*A\* document defining the general state of the art which is not considered to be of particular relevance

\*E\* earlier document but published on or after the international filing date

\*L\* document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

\*O\* document referring to an oral disclosure, use, exhibition or other means

\*P\* document published prior to the international filing date but later than the priority date claimed

\*T\* later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

\*X\* document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

\*Y\* document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

\*&amp;\* document member of the same patent family

Date of the actual completion of the international search

4 May 2005

Date of mailing of the international search report

25/05/2005

Name and mailing address of the ISA

European Patent Office, P.B. 5818 Patentlaan 2  
NL - 2280 HV Rijswijk  
Tel. (+31-70) 340-2040, Tx. 31 651 epo nl,  
Fax: (+31-70) 340-3016

Authorized officer

Kamps, S

## INTERNATIONAL SEARCH REPORT

International Application No

PCT/EP2005/050500

## C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X,P	<p>WIEGAND C, SIEMERS C, RICHTER H:  "Definition of a Configurable Architecture  for Implementation of Global Cellular  Automaton"  LECTURE NOTES IN COMPUTER SCIENCE,  'Online!  vol. 2981, February 2004 (2004-02), pages  140-155, XP002326434  ISSN: 0302-9743  ISBN: 3-540-21238-8  Retrieved from the Internet:  URL:http://www.springerlink.com/media/16GP  UPQTM0U7JAMNGADT/Contributions/6/L/U/6/6LU  6M1C0YCR9MEV2.pdf&gt;  'retrieved on 2005-04-28!  the whole document</p>	1-9
Y	<p>JOZWIAK L ET AL: "Effective and efficient  FPGA synthesis through general functional  decomposition"  JOURNAL OF SYSTEMS ARCHITECTURE, ELSEVIER  SCIENCE PUBLISHERS BV., AMSTERDAM, NL,  vol. 49, no. 4-6,  September 2003 (2003-09), pages 247-265,  XP004464564  ISSN: 1383-7621</p>	3
A	<p>page 248; figure 2; table 1</p>	4-6
Y	<p>MIRSALEHI M M, GAYLORD T K: "Logical  minimization of multilevel coded  functions"  APPLIED OPTICS,  vol. 25, no. 18,  15 September 1986 (1986-09-15), pages  3078-3088, XP002326433</p>	7
A	<p>page 3078 - page 3079  page 3081 - page 3083</p>	4-6
A	<p>HOFFMANN R, VÖLKMANN K P, HEENES W:  "Globaler Zellularautomat (CGA): Ein neues  massivparalleles Berechnungsmodell"  PARS WORKSHOP, October 2001 (2001-10),  XP002326432  MUNICH  the whole document</p>	1-9
A	<p>HOFFMANN R ET AL: "GCA: a massively  parallel model"  PARALLEL AND DISTRIBUTED PROCESSING  SYMPOSIUM, 2003. PROCEEDINGS.  INTERNATIONAL APRIL 22-26, 2003,  PISCATAWAY, NJ, USA,IEEE,  22 April 2003 (2003-04-22), pages 270-276,  XP010645542  ISBN: 0-7695-1926-1  the whole document</p>	1-9

A. KLASSIFIZIERUNG DES ANMELDUNGSGEGENSTANDES  
IPK 7 G06F15/80 G06F17/50

Nach der Internationalen Patentklassifikation (IPK) oder nach der nationalen Klassifikation und der IPK

**B. RECHERCHIERTE GEBIETE**

Recherchierter Mindestprüfstoff (Klassifikationssystem und Klassifikationssymbole)  
IPK 7 G06F

Recherchierte aber nicht zum Mindestprüfstoff gehörende Veröffentlichungen, soweit diese unter die recherchierten Gebiete fallen

Während der internationalen Recherche konsultierte elektronische Datenbank (Name der Datenbank und evtl. verwendete Suchbegriffe)

EPO-Internal, INSPEC

**C. ALS WESENTLICH ANGESEHENE UNTERLAGEN**

Kategorie*	Bezeichnung der Veröffentlichung, soweit erforderlich unter Angabe der in Betracht kommenden Teile	Betr. Anspruch Nr.
X	RECHENBERG, POMBERGER: "Informatik-Handbuch, 3. Auflage"	1, 2, 8, 9
Y	2002, HANSER, MÜNCHEN, XP002327006	3, 7
A	Seite 275  ----- -/--	4-6

☒ Weitere Veröffentlichungen sind der Fortsetzung von Feld C zu entnehmen

☐ Siehe Anhang Patentfamilie

\* Besondere Kategorien von angegebenen Veröffentlichungen :

\*A\* Veröffentlichung, die den allgemeinen Stand der Technik definiert, aber nicht als besonders bedeutsam anzusehen ist

\*E\* älteres Dokument, das jedoch erst am oder nach dem internationalen Anmeldedatum veröffentlicht worden ist

\*L\* Veröffentlichung, die geeignet ist, einen Prioritätsanspruch zweifelhaft erscheinen zu lassen, oder durch die das Veröffentlichungsdatum einer anderen im Recherchenbericht genannten Veröffentlichung belegt werden soll oder die aus einem anderen besonderen Grund angegeben ist (wie ausgeführt)

\*O\* Veröffentlichung, die sich auf eine mündliche Offenbarung, eine Benutzung, eine Ausstellung oder andere Maßnahmen bezieht

\*P\* Veröffentlichung, die vor dem internationalen Anmeldedatum, aber nach dem beanspruchten Prioritätsdatum veröffentlicht worden ist

\*T\* Spätere Veröffentlichung, die nach dem internationalen Anmeldedatum oder dem Prioritätsdatum veröffentlicht worden ist und mit der Anmeldung nicht kollidiert, sondern nur zum Verständnis des der Erfindung zugrundeliegenden Prinzips oder der ihr zugrundeliegenden Theorie angegeben ist

\*X\* Veröffentlichung von besonderer Bedeutung; die beanspruchte Erfindung kann allein aufgrund dieser Veröffentlichung nicht als neu oder auf erfinderischer Tätigkeit beruhend betrachtet werden

\*Y\* Veröffentlichung von besonderer Bedeutung; die beanspruchte Erfindung kann nicht als auf erfinderischer Tätigkeit beruhend betrachtet werden, wenn die Veröffentlichung mit einer oder mehreren anderen Veröffentlichungen dieser Kategorie in Verbindung gebracht wird und diese Verbindung für einen Fachmann naheliegend ist

\*&\* Veröffentlichung, die Mitglied derselben Patentfamilie ist

Datum des Abschlusses der internationalen Recherche

4. Mai 2005

Absendedatum des internationalen Recherchenberichts

25/05/2005

Name und Postanschrift der Internationalen Recherchenbehörde  
Europäisches Patentamt, P.B. 5818 Patentlaan 2  
NL - 2280 HV Rijswijk  
Tel. (+31-70) 340-2040, Tx. 31 651 epo nl,  
Fax: (+31-70) 340-3016

Bevollmächtigter Bediensteter

Kamps, S

C.(Fortsetzung) ALS WESENTLICH ANGESEHENE UNTERLAGEN		
Kategorie°	Bezeichnung der Veröffentlichung, soweit erforderlich unter Angabe der in Betracht kommenden Teile	Betr. Anspruch Nr.
X,P	<p>WIEGAND C, SIEMERS C, RICHTER-H:            "Definition of a Configurable Architecture            for Implementation of Global Cellular            Automaton"            LECTURE NOTES IN COMPUTER SCIENCE,            'Online!            Bd. 2981, Februar 2004 (2004-02), Seiten            140-155, XP002326434            ISSN: 0302-9743            ISBN: 3-540-21238-8            Gefunden im Internet:            URL: <a href="http://www.springerlink.com/media/16GPUPQTM0U7JAMNGADT/Contributions/6/L/U/6/6LU6M1C0YCR9MEV2.pdf">http://www.springerlink.com/media/16GPUPQTM0U7JAMNGADT/Contributions/6/L/U/6/6LU6M1C0YCR9MEV2.pdf</a>            'gefunden am 2005-04-28!            das ganze Dokument</p>	1-9
Y	<p>JOZWIAK L ET AL: "Effective and efficient            FPGA synthesis through general functional            decomposition"            JOURNAL OF SYSTEMS ARCHITECTURE, ELSEVIER            SCIENCE PUBLISHERS BV., AMSTERDAM, NL,            Bd. 49, Nr. 4-6, September 2003 (2003-09),            Seiten 247-265, XP004464564            ISSN: 1383-7621</p>	3
A	<p>Seite 248; Abbildung 2; Tabelle 1</p>	4-6
Y	<p>MIRSALEHI M M, GAYLORD T K: "Logical            minimization of multilevel coded            functions"            APPLIED OPTICS,            Bd. 25, Nr. 18,            15. September 1986 (1986-09-15), Seiten            3078-3088, XP002326433</p>	7
A	<p>Seite 3078 - Seite 3079            Seite 3081 - Seite 3083</p>	4-6
A	<p>HOFFMANN R, VÖLKMANN K P, HEENES W:            "Globaler Zellularautomat (CGA): Ein neues            massivparalleles Berechnungsmodell"            PARS WORKSHOP, Oktober 2001 (2001-10),            XP002326432            MUNICH            das ganze Dokument</p>	1-9
A	<p>HOFFMANN R ET AL: "GCA: a massively            parallel model"            PARALLEL AND DISTRIBUTED PROCESSING            SYMPOSIUM, 2003. PROCEEDINGS.            INTERNATIONAL APRIL 22-26, 2003,            PISCATAWAY, NJ, USA, IEEE,            22. April 2003 (2003-04-22), Seiten            270-276, XP010645542            ISBN: 0-7695-1926-1            das ganze Dokument</p>	1-9